EXHIBIT A

Microsoft Patent Pre-disclosure Document

Note: Hover the pointer over underlined section headings for more help and click on a link for a sample document.

[redacted]

Introduction:

This patent is about the ability for an application to decide which features are so important that they should be available on screen even when the application is minimized. It also covers the users' ability to choose which applications do this, where the UI is placed, and how much room they take up.

Strategic Importance:

Many applications have some set of features that are useful to a user even when the application window is minimized. Some applications offer mini versions so that a user can make the app very small and out of the way while still having access to important information and controls.

With Rich Minimized Apps that live in the Sidebar we have a standard way for applications to provide controls and information even when they're minimized. The user has control over which applications are able to do this, and how much room they take up when they're minimized.

Description of the invention:

A rich minimized app is an application that moves into the sidebar when it is minimized. These are a type of transient tile, and so they follow all the insertion rules for other transient tiles (see UI Squishing)

An application (such as a media player) that wants to offer controls and information to the user even when it's minimized (such as song name, and play/pause controls) can build a rich minimized app tile for the sidebar. When the user minimizes the application by clicking on the minimize button (or using another way of minimizing a window), the main application window goes away and a tile appears in the sidebar. The tile provides some information and controls that were in the main window so that the user still has access to them even though the window is minimized.

When the user wants to switch back to the larger window, they click on the restore button in the tile. The restore button is unique to RMA tiles. This re-opens the application window and removes the tile from the sidebar.

The sidebar provides several functions for an application that provides an RMA. When the app minimizes, the sidebar will draw the animation zooming to the sidebar rather than the taskbar. The sidebar will remove the window's button from the taskbar and will remove the entry from the Alt-tab menu. When the window is restored, the sidebar will ensure the the taskbar button and Alt-tab entry show again.

Special controls

1. An RMA header has a restore button that appears next to the icon on hover
2. The right-click menu of the RMA tiles has Restore, Maximize, Close.
3. The "Remove from bar" option has a confirmation dialog so that it isn't confused with close.
   - 3.1. The dialog says "Removing this tile will prevent this app from minimizing to the Sidebar in the future. Ok Cancel"

Standard behavior

Aside from its special controls, a minimized app should act like any other transient tile.

[redacted]

After a reboot or logout a minimized app should remain in the bar (though if this is too difficult it is not required). The RMA tile is responsible for restarting any processes that are required to make it work, and the RMA must also tell the bar what app to launch when it's restored.

Spam prevention

If a user does not want an app to minimize to the sidebar, they can select the "Remove from bar" option in the context menu or Add / Remove tiles page. This will prevent the app from minimizing to the bar. The user gets a confirmation in this case to ensure they didn't mean to just close the tile. The user can re-add it from the Add/Remove page and it will again be allowed to minimize to the bar.

Implementation

1. An app that wants to minimize to the sidebar calls **SidebarMinimizedApp** with 3 paramaters: the **guid of the tile**, an **init string** to get the tile started, and a **pointer to its hwnd** (optional.
   - 1.1. The sidebar watches to see when the window is minimized
   - 1.2. When it's minimized (or if there was no hwnd to begin with), the sidebar hides the app's window, taskbar button, and alt-tab entry, and shows the tile in the bar
2. When the user clicks the restore button in the tile
   - 2.1. If the tile **has implemented a Restore handler**,
     - 2.1.1.The sidebar calls it on a separate thread
     - 2.1.2.The tile UI is hidden
     - 2.1.3.When the handler returns, the sidebar disposes of the tile on the original thread
   - 2.2. If the tile **has not** implemented a Restore handler,
     - 2.2.1.The tile UI is hidden
     - 2.2.2.The sidebar disposes the tile
     - 2.2.3.The sidebar restores the app window, taskbar button, and alt-tab entry

[redacted]